Pattern Anal Applic (2003) 6: 232–244
DOI 10.1007/s10044-003-0191-0

ORIGINAL ARTICLE

S.-W. Kim, B. John Oommen

A brief taxonomy and ranking of *creative* prototype reduction schemes

Received: 6 June 2002 / Accepted: 26 March 2003
© Springer-Verlag London Limited 2003

Abstract Various Prototype Reduction Schemes (PRS) have been reported in the literature. Based on their operating characteristics, these schemes fall into two fairly distinct categories — those which are of a *creative* sort, and those which are essentially *selective*. The norms for evaluating these methods are typically, the reduction rate and the classification accuracy. It is generally believed that the former class of methods is superior to the latter. In this paper, we report the results of executing various creative PRSs, and attempt to comparatively quantify their capabilities. The paper presents a brief taxonomy of the various reported PRS schemes. Our experimental results for three artificial data sets, and for samples involving real-life data sets, demonstrate that no *single* method is uniformly superior to the others for all kinds of applications. This result, though consistent with the findings of Bezdek and Kuncheva [1], is, in one sense, counter-intuitive, because the various researchers have presented their specific PRS with the hope that it would be superior to the previously reported methods. However, the fact is that while one method is superior in certain domains, it is inferior to another method when dealing with a data set with markedly different characteristics. The conclusion of this study is that the question of determining *when* one method is superior to another remains open. Indeed, it appears as if the designers of the pattern recognition system will have to choose the appropriate PRS based to the *specific* characteristics of the data that they are studying. The paper also suggests answers to various hypotheses that relate to the accuracies and reduction rates of families of PRS.

Keywords Machine Learning · Prototype Reduction Schemes (PRS) · Learning Vector Quantisation (LVQ) · Support Vector Machines (SVM)

Introduction

Currently, huge amounts of multimedia information is available on the Internet. Classifying, understanding or compressing this information is a difficult task. The same problem is also associated with extremely large data sets such as those encountered in data mining, text categorisation, financial forecasting, retrieval of multimedia databases and biometrics. One possible solution to this problem is to reduce the number of sample vectors, while simultaneously insisting that the decisions based on the reduced data set perform as well, or nearly as well, as the decisions based on the original data set. This idea has been explored by many researchers, and this has resulted in the development of many algorithms [1,2].

It is interesting to note that Bezdek et al [1], who composed an excellent survey of the field, report that there are 'zillions!' of methods for finding prototypes ([1], p. 1459]). Our work does not compete with theirs — it merely supplements their results with an ensemble of experiments performed with a few of the creative (replacement) schemes for a variety of data sets. Rather than re-embark on a brand new survey of the field, we briefly 'sample' a *few* representative methods of the 'zillions' that have been reported. However, we emphasise that even this will not be done in an exhaustive and comprehensive manner in this relatively brief paper. Rather, although we catalogue an ensemble of the various *creative* PRS, we shall explain in greater detail, only those *creative prototype reduction schemes* that we have specifically used in the comparison.[1]

*Sang-Woon Kim (✉)
Division of Computer Science and Engineering, Myongji University, Yongin, 449-728 Korea.
E-mail: kimsw@mju.ac.kr

B. John Oommen
School of Computer Science, Carleton University, Ottawa, Canada K1S 5B6.
E-mail: oommen@scs.carleton.ca.
‡Partially supported by the Natural Sciences and Engineering Research Council of Canada. A preliminary version of this paper was presented at the 2002 IEEE SMC Conference in October 2002 in Hammamet, Tunisia.

[1] In this regard, we thank the anonymous referees for suggesting this strategy.

One of the first of its kind was the Condensed Nearest Neighbor (CNN) rule [3]. This rule, however, includes 'interior' samples which can be eliminated completely without inviting changes in the performance. Results relating to the CNN are mentioned here so as to serve as a benchmark.

Since the development of the CNN, other methods have been proposed successively, such as the Reduced Nearest Neighbour (RNN) rule [4], the Prototypes for Nearest Neighbor (PNN)[2] classifiers [5], the Selective Nearest Neighbour (SNN) rule [6], two modifications of the CNN [7], the Edited Nearest Neighbour (ENN) rule [8], and the Parzen approach to non-parametric data reduction [9]. Besides these methods, the Vector Quantisation (VQ) technique [10] and the Bootstrap Technique (BT) [11] have also been reported to be extremely effective approaches in data reduction problems. Additionally, a formulation of Learning Vector Quantisation (LVQ) [12] combined with the stimulated annealing algorithm [13] has been shown to generate optimal reference models of large data set problems. Recently, the clustering-based methods using Genetic Algorithms (GA) and Random Search (RS) [15], Proximity-Graph-based (PG) editing and condensing method [16], and the method of prototype selection based on the properties of Polyline Functions (PF) [17], have also been investigated, independently. Similarly, Chen and Jowick [20] use neural network principles (as opposed to 'pure' pattern recognition principles) to reduce the size of the training set without significantly decreasing the classification quality. Although the effectiveness of their proposed algorithm is compared with that of the Class Sensitive Neural Network (CSNN) presented by Chen and You in 1993, Chen and Jowick [20] argue that the same approach can also be applied to other kinds of classifiers. The work of Chen and Jowick [20] has been verified by testing it on a very large remote-sensing data set.

Apart from the papers that are mentioned in greater detail, we feel that it is appropriate to take a few minutes to highlight a new PRS introduced in Wu et al [21]. Wu et al [21] propose a new *principle* by which the template size can be considerably reduced. This principle is derived from the fact that if many prototypes of the same class are found in any area of the feature space, and if this area does not contain any prototypes from any of the *other* classes, an unknown pattern would be correctly classified to be of this particular class[3] Wu et al [21] invoke this idea to reduce redundant prototypes of any one class, whose samples fall within this specified area of interest. Observe now that if an unknown pattern belonging to this class is located in this area of interest, then the number of patterns in this area from the correct class, which are the nearest neighbours to the unknown pattern, need not declare the unknown pattern's identity by *unanimity*. Rather, this could be achieved by just a *majority* vote, and consequently, some of the prototypes of this class which fall within the area, can be discarded without *any* reduction in accuracy. This concept differentiates the work of Wu et al [21] from the other PRS.

On the other hand, Support Vector Machines (SVM) [18] have the capability of extracting vectors that support the boundary between the two classes, and they thus satisfactorily represent the global distribution structure. As opposed to this, the LVQ has a capability of adjusting the prototypes (code-book vectors) so that the decision boundaries for every pair of neighbouring classes satisfy the approximation of the class distribution. Thus, apart from the above methods, the SVM can also be used as a means of selecting prototype vectors, which are subsequently adjusted by means of an LVQ3-type method. Such a novel prototype reduction method obtained by hybridising the SVM and the LVQ3 has been recently reported in Kim and Oommen [19].

As is obvious from the above catalogue of methods, the researcher is left with the sense that the number of methods is large, and with the question of resolving which methods s/he has to use for any particular application data set. We endeavor to simplify his/her task by first categorising the methods using a simple taxonomy, and then quantifying their relative capabilities by testing them on an ensemble of benchmark data-sets.

The above approaches can be categorised into two groups by considering their operating characteristics. First, they can be classified as being either *creative* or *selective*, based on whether they *create new* prototypes, or they, rather, merely *select* some of the existing data points as the prototypes. From another perspective, the algorithms can be classified as being either *deterministic* or *non-deterministic*, depending on whether or not we can *control* the number of prototypes generated by the algorithms.

All the reported schemes are required to reduce the samples of the input data set to a small number of representatives that can represent the entire set satisfactorily. Thus, their performance should be evaluated in terms of both the classification accuracy and the reduction rates. In this paper, we conduct an empirical comparison between the various *creative* schemes (including the hybrid method) for three artificial and real-life data sets. The comparisons are made from the point of view of their data reduction rates, the classification accuracy rates and computational efficiencies.

The paper is organised as follows. In Sect. 2, we provide a simplified taxonomy of the representative prototype reduction methods. Section 3 provides an experimental comparison and discussions on the creative prototype

[2] Bezdek and his co-authors, proposed a modification of the PNN [7]. First, instead of using the *weighted* mean of the PNN to merge prototypes, they utilised the simple arithmetic mean. Secondly, the process for searching for the candidates to be merged was modified by partitioning the distance matrix into submatrices 'blocked' by common labels. This modification eliminated the consideration of candidate pairs with different labels.

[3] This is, of course, the underlying principle of the nearest neighbour method, and also of all the non-parametric methods for density estimation.

reduction schemes for both artificial data sets and real-life data sets. Finally, the conclusions are given in Sect. 4.

Contributions of the paper

The main contribution of this paper is the formal, objective and systematic study of the various families of *creative* PRS. Since each researcher attempts to promote his *particular* paradigm, the literature contains a catalogue of such methods, and various experimental results that arguably demonstrate the superiority of one scheme over the other. The first objective of the paper is to submit a simplified taxonomy of the various PRS available. The primary contribution of this paper is the assertion that there seems to be no clear scheme that is *uniformly* superior to all other PRS. This has been clearly demonstrated in experiments involving both synthetic and real-life data. Instead, it appears as if each method has its distinct advantages. Furthermore, the question of determining *when* one method is superior to another, is, generally speaking, *open*. We believe that this really depends upon the data, its clustering patterns, the outliers of the class distributions, the number of data points themselves, and of course, the proximity relationships between the individual classes.

The results[4] presented here complement the results of the study of Bezdek and Kuncheva [1]. They also form a basis by which some accuracy/reduction-rate hypotheses can be verified. Finally, they can also be used as a benchmark against which future research can be 'calibrated'.

A taxonomy of PRS

As mentioned previously, various data reduction methods have been proposed in the literature — two excellent surveys are found in Dasarathy [2] and Bezdek and Kuncheva [1]. The survey of Bezdek and Kuncheva [1] contains a comparison of eleven conventional PRS methods: a combination of Wilson's ENN and Hart's CNN (W+H); a Genetic Algorithms (GA) and Random Selection (RS) method; a Tabu Search (TS) scheme; a Vector Quantisation-based method (LVQ1); a Decision Surface Mapping (DSM); a scheme which involves LVQ with Training Counters (LVQTC); a Bootstrap (BTS) method; a Vector Quantisation (VQ) method; a Generalised LVQ-Fuzzy (GLVQ-F) scheme; and a Hard C-Means clustering (HCM) procedure.

Among these, the W+H, RS, GA and TS can be seen to be selective PRS schemes, and the others fall into the category of being creative. Additionally, the VQ, GLVQ-F and HCM are post-supervised approaches in which the methods first find prototypes without regard to the training data labels, and then assign a class label to each prototype, while the remaining are pre-supervised ones that use the data and the class labels together to find the prototypes. Finally, the RS, LVQ1, DSM, BT, VQ and GLVQ-F are capable of permitting the user to define the number of prototypes, while the rest of the schemes force the algorithm to decide this number.

With regard to the taxonomy, first, the PRS can be categorised into two groups by considering whether or not they can *create new* prototypes, or whether they, rather, merely *select* some of the existing data points as the prototypes. The schemes reported by Chang [5] (the PNN)[5] Xie et al [10] (the VQ), Hamamoto et al [11] (the BT), Song and Lee [13] (the LVQ) and Kim and Oommen [19] (HYB), *create* new prototype vectors (and do not merely select training samples) in such a way that these prototypes represent all the vectors in the original set in the 'best' possible manner. The methods of the Hart [3] (the CNN), Gates [4] (the RNN), Ritter et al [6] (the SNN), Tomek [7] (the mCNN), Devijer and Kittler [8] (the ENN), Fukunaga [9] (the PZN), Bezdek [15] (the GA and RS), Sanchez et al [19] (the PG) and Lipowezky [17] (the PF) are those in which the prototype vectors are merely *selected*. It has been proven that the former family is partially superior to the latter [1,10].

Basically, all reduction algorithms repeat the selecting (or creating) process until the desired reduced prototypes are obtained. Therefore, the PRS can also be classified into two categories based on their selecting criteria, and on the conditions when the processes terminate. In some cases, the user can estimate the number of processing iterations by considering the number of sample (Bootstrap) vectors or the code-book size. In other cases, we cannot estimate *a priori* the maximum number of iterations which should be done. The former is the criterion employed in the CNN [3], RNN [4], SNN [6], mCNN [7], ENN [8], VQ [10], BT [11] and GA (RS) [15], and the latter is the criterion used in the PNN [5], PZN [9], LVQ [13], the PG [16] and HYB [19][6] In any PRS, the number of final prototype vectors is an important factor for determining the compression rate. So, the PRS can also be classified according to whether or not we can estimate the number of final prototype vectors in advance. In the approaches

[4] It should be particularly mentioned that it is not feasible to survey and implement *every* single instantiation of every reported method (and all the variants) in this paper. Rather, we have chosen to implement a *representative* instantiation of each method examined, and used it as a basis for comparison. However, we should add that, to be fair, this comparison has been made on a level playing field, by reporting the best results over the permitted parameter space applicable for the method under consideration.

[5] In this context, it is also fitting to include a modified Chang's method proposed by Bezdek et al [14].

[6] Our earlier paper [19] had presented a *Hybrid* scheme. In that paper, some comparative results were also given. This present paper contains a more complete record of the experimental results (including the results for the 'Non—normal' data set), and also compares the results with methods which are not compared in Kim and Oommen [19].

Table 1 A simplified taxonomy on the prototype reduction schemes useful for obtaining a smaller prototype set

Criterion	Types	The PRS methods
Creation/ Selection	Creative	PNN, VQ, BT, LVQ, HYB
	Selective	CNN, RNN, SNN, mCNN, ENN, PZN, GA (RS), PG, PF
# of Iterations	Predetermined	CNN, RNN, SNN, mCNN, ENN, VQ, BT, GA (RS)
	Data Dependent	PNN, PZN, LVQ, HYB, PG
# of Prototypes	Controllable	VQ, BT
	Uncontrollable	CNN, RNN, PNN, SNN, mCNN ENN, PZN, GA (RS), HYB, PG, PF

of the VQ [10] and BT [11], we can decide the number of prototype vectors based on the control strategy of the algorithms. On the other hand, in the case of the CNN [3], RNN [4], PNN [5], SNN [6], mCNN [7], ENN [8], PZN [9], GA (RS) [15], PG [16], PF [17] and HYB [19], the number of prototype vectors is itself a random variable, and we cannot determine it until the processes are finished. For some applications, this feature of the PRS is often more significant than achieving a higher classification accuracy.

Based on these three criteria[7] a taxonomy of the reported PRS is summarised in Table 1. In this connection, it is pertinent to mention the following. The classification based on the third criterion (i.e. whether the number of iterations of PRS is predetermined or data-dependent) is not a clear-cut criterion, and hence the tabulation of the methods that we propose is debatable (or rather, subjective). For example, the CNN does not have a predetermined number of iterations, as stated in the original paper. Also, methods which use GA principles can be classified into either of the two categories depending on the actual instantiations. With regard to the final number of prototypes, we believe that the LVQ and VQ algorithms are to be placed in different subgroups. This too is arguable. However, when we mention the instantiations of these two, we specifically consider the LVQ scheme in which the number of prototypes is not fixed *a priori*, and the VQ scheme in which the number of prototypes is fixed *a priori*. Finally, the question of whether the number of iterations used is predetermined or data-dependent, is really not a characteristic of the algorithm, but is a matter of choice by the designer. However, we have chosen to use it as a criterion, so that the reader knows that this is a valid option by which the accuracy (time) can be controlled.

[7] The first and the second criteria employed for the taxonomy were used earlier in Bezdek and Kuncheva [1]. The third criterion is fairly subjective, as we shall see presently. However, as we argue later, the real problem which is to be considered is to find a single criterion involving the accuracy, the time of computation, the reduced number of prototypes, and the features of the data sets, which will *collectively* inform the user of the best PRS to be used. We shall present the complexity of attempting such a strategy in a later section. We are grateful to the anonymous Referees who suggested these changes.

Prototype reduction schemes (PRS)

As mentioned previously, various data reduction methods have been proposed in the literature — a survey of which is found in Dasarathy [2] and Bezdek and Kuncheva [1]. The *creative* PRS are reviewed here. The reviews are necessarily brief, and must not be reckoned to be either exhaustive or comprehensive. Rather, although we catalogue an ensemble of the various *creative* PRS, in this section, we explain in greater detail, only those *creative* PRS that we have specifically used in the comparison.

The condensed nearest neighbour rule (CNN)

The CNN [3] is suggested as a rule which reduces the size of the design set, and is largely based on statistical considerations. However, the rule does not, in general, lead to a minimal consistent set — a set which contains a minimum number of samples within it to correctly classify all the remaining samples in the given set. The procedure can be formalised as follows, where the training set is given by T, and reduced prototypes are found in T_{cnn}.

1. The first sample is copied from T to T_{cnn};
2. Do the following: increasing i by unity from 1 to the number of samples in T per epoch:
   (a) Classify each pattern $x_i \in T$ using T_{cnn} as the prototype set;
   (b) If a pattern x_i is classified incorrectly then add the pattern to T_{cnn}, and go to (3);
3. If i is not equal to the number of samples in T, then go to (2);
4. Else the process terminates.

Prototypes for nearest neighbour (PNN) classifiers

The algorithm of finding Prototypes for Nearest Neighbour classifiers (referred to as PNN, here) [5] can be stated as follows: given a training set T, the algorithm starts with every point in T as a prototype. Initially, set A is empty and set B is equal to T. The algorithm selects an arbitrary point in B and initially assign it to A. After this, the two closest prototypes p in A and q in B of the same class are merged, successively, into a new prototype, p^*, if the merging will not degrade the classification of the patterns in T, where p^* is the weighted average of p and q. For example, if p and q are associated with weights W_p and W_q, respectively, p^* is defined as $(W_p \cdot p + W_q \cdot q)/(W_p + W)$, and is assigned a weight, $W_p + W_q$. Initially, every prototype has an associated weight of unity. The procedure of PNN is sketched below.

1. Copy T to B;

2. For all q ($\in B$, set the weight $W_q = 1$;
3. Select a point in B, and move it from B to A;
4. MERGE = 0;
5. While B is not empty do:
   (a) Find the closest prototypes p and q from A and B, respectively;
   (b) If p's class is not equal to q's class then insert q to A and delete it from B;
   (c) Else merge p of weight W_p, and q of weight W_q., to yield p^*, where $p^* = (W_p \cdot p + W_q \cdot q)/(W_p + W_q)$. Let the classification error rate of this new set of prototypes be ϵ;
   If the ϵ is increased then insert q to A, and delete it from B;
   Else delete p and q from A and B, insert p^* with weight $W_p + W_q$ to A, and MERGE++;
6. If MERGE is equal to 0 then output A as the set of trained code-book vectors, and the process terminates;
7. Copy A into B and go to 3.

Bezdek and his co-authors, proposed a modification of the PNN [14]. First, instead of using the *weighted* mean of the PNN to merge prototypes, they utilised the simple arithmetic mean. Secondly, the process for searching for the candidates to be merged was modified by partitioning the distance matrix into submatrices 'blocked' by common labels. This modification eliminated the consideration of candidate pairs with different labels. Based on the results obtained from experiments conducted on the Iris data set, the authors of Bezdek et al [14] asserted that their modified form of the PNN yielded the best consistent reduced set for designing multiple-prototype classifiers[8].

Vector quantisation (VQ) techniques

The foundational ideas motivating VQ (Vector Quantisation) and the SOM (Self Organising Map) are the classical concepts that have been applied in the estimation of probability density functions. The concept of VQ or SOM can be perceived as one of the prototype reduction approaches. Rather than represent the entire data in a compressed form using only the estimates, VQ opts to represent the data in the actual feature space. The vector quantisation technique for non-parametric data reduction [10] is as follows:

1. For each class i, ($i = 1, \ldots, N$), find the M_i level optimal quantisers using a VQ algorithm [23]. Suppose the final quantiser reproduction symbols are $A_i = \{y_j^i;\ j = 1, \ldots, M_i\}$ ($i = 1, \ldots, N$);
2. Combine all the reproduction alphabets A_i of all classes into one single set A of M vectors, that is, $U = \bigcup_i A_i$ and $M = \sum_i M_i$. Output A as the set of final prototypes.

The boostrap techniques (BT) for nearest neighbour classifiers

The bootstrap technique, which has been applied for the small training sample size situations is also used for reducing the prototypes. This technique can also be effectively used to perform data reduction in the design of the 1-NN classifier. The bootstrapping method [11] is briefly sketched below:

1. For every class, i, select a sample x_i from T_{N_i} in random;
2. Find the k nearest neighbour samples of the x_i as x_i^1, $x_i^2, \ldots, x_i^k$;
3. Compute a bootstrap sample (prototype) $y_i = \frac{1}{k}\sum_{j=1}^{k} x_i^j$;
4. Repeat the above three steps $M_i (M_i \leq N_i)$ times, under a condition that no sample is selected more than once;
5. Combine all the bootstrap samples y_i of all classes into one single set of M vectors, and output it as the set of final prototypes.

It has been believed that as the number of prototypes decreases, the error increases. However, the results of the experiments in Hamamoto et al [11] contradict the above hypothesis, and suggest that we may need only a relatively small number of prototypes.

The hybrid—Kim—Oommen algorithm

The Hybrid—Kim—Oommen algorithm [19] essentially consists of two steps. First, initial prototypes are *selected* or *created* by any of the conventional reduction methods described earlier. After this selection/creation phase, we invoke a phase in which the optimal positions are learned with an LVQ3-type scheme. The procedure is formalised below for each class:

1. For every class, j, select an initial condensed prototype set $Y_{j,Test}$ by using any one of the reduction methods described earlier, and the entire training sets, $T_{i,t}$;
2. Using $Y_{j,Test}$ as the set of condensed prototype vectors for class j, do the following using the *Placement* sets, $T_{i,p}$, and the Optimising sets, $T_{i,0}$ for all the classes:
   (a) Perform LVQ3 using the points in the *Placement* set, $T_{i,P}$. The parameters of the LVQ3 are spanned by considering increasing values of w from 0.0 to 0.5, in steps of Δw. The sets $Y_{j,Test}$ (for all j) and Y_{Test} are updated in the process. Select the best value w_0 after evaluating the accuracy of the classification rule on $T_{i,0}$, where the NN-classification is achieved by the adjusted Y_{Test};
   (b) Perform LVQ3 using the points in the *Placement*

[8] We believe that the LVQ3-based enhancement that we proposed in Kim and Oommen [19] can also be used to enhance the scheme proposed in Bezdek et al [14]. We also believe that a similar enhancement can be used for the clustering-based, genetic and random search methods proposed in Kuncheva and Bezdek [15]. This is currently being investigated. The authors are grateful to Professor Jim Bezdek for the instructive discussions we had in Spain in April 2002.

set, $T_{i,P}$. The parameters of the LVQ3 are again spanned by considering increasing values of ϵ from 0.0 to 0.5, in steps of $\Delta\epsilon$. The sets $Y_{j,Test}$ (for all j) and Y_{Test} are updated in the process. Select the best value ϵ_0 after evaluating the accuracy of the classification rule on $T_{i,O}$, where the NN-classification is achieved by the adjusted Y_{Test};

(c) Repeat the above steps with the current w_0 and ϵ_0, till the best values w^* and ϵ^* are obtained;

3. Determine the best prototype set Y_{Final} by invoking LVQ3, η times, with the data in $T_{i,P}$, and where the parameters are w^* and ϵ^*. Again, the 'pseudo-testing' is achieved using the Optimising set, $T_{i,O}$.

The actual classification accuracy is obtained by testing the classifier using the final values Y_{Final} and the original testing (validation) data points, $T_{i,V}$. The details of the scheme are omitted here — they can be found in Kim and Oommen [19], which also reports the results of the scheme for both artificial and real-life data sets.

Experiments

The survey of Bezdek and Kuncheva [1] contains a comparison of 11 conventional PRS methods. This comparison has been performed from the view of error rates, and of the resultant number of prototypes that are obtained. The experiments were conducted with four experimental data sets which are both artificial and real.

Based on the experimental results obtained, the authors of Bezdek and Kuncheva [1] claims that there seems to be no clear scheme that is *uniformly* superior to all the other PRS. Indeed, different methods were found to be superior for different data sets. However, the experiments showed that the *creative* methods are generally superior to the selective methods, but are, typically, computationally more difficult to determine.

In this paper, we report the results of more exhaustive experiments, and provide a comparison with the most pertinent *creative* PRSs, which are the PNN, the VQ, the BT, and the HYB. Also, to render the comparison complete, we compare them with a conventional method, the CNN, which is chosen as a representative scheme of the *selective* methods.

Experimental data

The creative PRSs were evaluated and compared by performing experiments on a number of design data sets, both real-life and artificial, as summarised in Table 2[9]

The dataset described as 'Random (in short, Rand)', was generated randomly with a uniform distribution, but with irregular decision boundaries. This is the same data set as was used earlier in Hart [3]. The dataset named as 'Non—normal (in short, Non—n)', which has also been used elsewhere [9–11], was generated from a mixture of four 8-dimensional Gaussian distributions as follows:

$$p_1(x) = \frac{1}{2}N(\mu_{11},I_8) + \frac{1}{2}N(\mu_{12},I_8)$$
$$p_2(x) = \frac{1}{2}N(\mu_{21},I_8) + \frac{1}{2}N(\mu_{22},I_8),$$

where μ_{11} = [0,0,…,0], μ_{12} = [6.58,0,…,0], μ_{21} = [3.29,0,…,0] and μ_{22} = [9.87,0,…,0]. Here, I_8 is the 8-dimensional *Identity* matrix.

The data set named 'Non—linear (in short, Non—l)', which has a strong non-linearity at its boundary, was generated artificially from a mixture of four variables as follows:

$$p_1(x) = \{x_1, \frac{1}{2}x_1^2 + y_1\},$$
$$p_2(x) = \{x_2, -\frac{1}{2}x_2^2 + y_2\}$$

where x_1, x_2, y_1, y_2 are normal random variables whose means and variances are (0, 10), (10, 5), (3, 10) and (20, 5), respectively. The total number of vectors per class is 500.

On the other hand, the datasets 'Iris2', 'Ionosphere (in short, Ionos)', 'Sonar', 'Arrhythmia' (in short, Arrhy), 'Glass' and 'Adult4', which are real benchmark data sets, are cited from the UCI Machine Learning Repository[10] Originally, the 'Iris' dataset consists of three classes : Setosa, Versicolor, and Virginica. However, since the subset of Setosa samples is completely separated from the others, it is not difficult to classify it from the other two. Therefore, we have opted to employ a modified set 'Iris2', which consisted only of the two classes, Versicolor and Virginica.

The 'Sonar' data set contains 208 vectors. Each sample vector, of two classes, has 60 attributes which are all continuous numerical values. The 'Arrhythmia' data set contains 279 attributes, 206 of which are real-valued and the rest are nominal. In our experiments, the nominal features were replaced by zeros. The aim of the pattern recognition exercise was to distinguish between the presence or absence of cardiac arrhythmia, and to classify the feature into one of the 16 groups. In our case, in the interest of uniformity, we merely attempted to classify the total instances into two category, namely, 'normal' and 'abnormal'.

The 'Glass' data set is a collection of glass fragments of different origin used for forensic investigations. Each sample vector, of two classes, has nine attributes, which are all continuous numerical values. Out of 214 total

[9] As seen from the surveys, numerous studies have been done with different data sets. Each of these data sets have had their own peculiar characteristics. When we were faced with the problem of determining which data sets were to be used, we opted to choose the benchmarks found in the well-acclaimed repositories. Also, to make the study more complete, we have included experimental results for both artificial and real-life data sets, both of which have been specifically ear-marked. But in both these cases,we have used only the standard data sets in the interest of repeatability.

[10] http://www.ics.uci.edu/mlearn/MLRepository.html

instances, the number of window glass (building windows and vehicle windows) instances is 163, and the number of non-window glass instances is 51.

The 'Adult4' data set was extracted from a census bureau database[11]. Each sample vector has fourteen attributes. Some of the attributes, such as the age, hours-per-week, etc., are continuous numerical values. The others, such as education, race, etc., are nominal symbols. In this case, the total number of sample vectors is 33,330. Among them, we selected randomly 8336 samples due to the time considerations, which is approximately 25% of the whole set.

In the above data sets, all of the vectors were normalised to be within the range $[-1,1]$ using their standard deviations, and the data set for class j was randomly split into two subsets, $T_{j,t}$ and $T_{j,v}$, of equal size. One of them was used for choosing the initial prototypes and training the classifiers, and the other one was used in their validation (or testing). Later, the role of these sets were interchanged.

Experimental parameters

As in the other learning algorithms, the parameters of the PRS also play an important role in determining the quality of the solution. The issue of determining the parameters for the creative PRS are summarised as follows:

1. Parameters for the PNN
   (a) None.
2. Parameters for the VQ
   (a) The code book size, M, is selected as one of the quantities $2^1,2^2,\ldots,2^p$ ($2^p \leq N$, N is the number of samples), after considering the classification accuracy,
   (b) The maximum number of iterations is 50,
   (c) The stopping and splitting criteria are both 0.01.
3. Parameters for the BT
   (a) The number of the bootstrap samples, M, is decided in the same way as in the case of the VQ's,
   (b) The number of nearest neighbor samples, k, for computing the bootstrap samples is chosen as one of the quantities $\frac{1}{4}N$, $\frac{2}{4}N$, $\frac{3}{4}N$,after considering the classification accuracy.
4. Parameters for the HYB
   (a) The initial code book size is determined by the SVM (in this experiment, we hybridised the SVM and an LVQ3 type algorithm),
   (b) The parameters for the LVQ3 learning, such as the α, the ϵ, the window length, w, and the iteration length, η, are specified as described in Kim and Oommen [19].

[11] http://www.census.gov/ftp/pub/DES/www/welcome.html

Experimental results

To evaluate the PRS, we first selected the prototypes from the training data sets using the CNN, the PNN, the VQ, the BT and the HYB algorithms. Subsequently, the test data sets were classified with the 1-NN classifiers designed with the selected prototypes. Finally, the experiments were repeated by exchanging the roles of the data sets. The prototype reduction rates, the classification accuracy rates and the computation time obtained from the experiments are shown in Tables 3, 4 and 5, respectively. Here, the values reported are the average values obtained after 10 trials. Observe that the first two lines for each data set are the results for the training and test subsets, respectively, and the third line (bold-faced) is the average of the above two lines. The values marked with '*' are to be ranked as the best two results among the five competing methods.

In Table 3, the index of comparison is the reduction rate, $Re(\cdot)$,computed as

$$Re(\cdot) = \frac{|Dataset| - |Prototypes|}{|Dataset|} \times 100(\%),$$

where $|\cdot|$ is the cardinality of the corresponding set. As opposed to this, in Table 4, the algorithms are compared based on their classification accuracy, $Acc(\cdot)$.

From Tables 3 and 4, we see that no specific method yields the best results for all the families of applications in terms of its reduction rate and classification accuracy. The best method for one data set is not the best for another data set. However, we can make some interesting observations. For high-dimensional applications such as the 'Sonar' (dimension, d=60) and 'Arrhythmia' (d=279) data sets, the PNN performs better than the others in terms of both the *Re* and *Acc* criteria. In these cases, the reduction rates are 67.79% and 96.68%, respectively, and the accuracies are 82.69% and 99.12%, respectively. The worst reduction rates for these data sets is obtained by the BT method where the reduction rate is merely 38.46% and 43.36%, respectively. On the other hand, for low-dimensional data sets such as the 'Random' (d=2) and 'Iris2' (d=4), the HYB performs better than the others in terms of both criteria. In these cases, the reduction rates for the sets are 90.5% and 85.0%, respectively, and the accuracies are 97.5% and 93.0%, respectively. The worst reduction rates for these data sets is obtained by the VQ method where the reduction rate is merely 36.0% and 68.0%, respectively. However, for middle-dimensional data sets such as the 'Non—normal' (d=8) and 'Ionosphere'(d=34), the VQ performs better than the others in terms of both criteria.

From Table 5, we see that the VQ is the fastest method. Also, the computation time of the PNN is generally speaking, larger than that of the others, and is, in particular, *much* more computationally intensive for the large-sized data set 'Adult4'. In the case of this data-set, it is more than three orders of magnitude slower than the VQ, and almost one order of magnitude slower than the HYB method. Such a comparative statement seems to be uni-

Table 2 The benchmark data sets for experiments. The vectors of each dataset are divided into two subsets of equal size, and used for training and validation, alternately. Also, in the interest of clarity, we have distinguished between the 'Artificial' and 'Real Life' data sets, respectively

Dataset Type	Dataset Names	# of Patterns	# of Features	# of Classes
	Random	400 (200,200)	2	2
Artificial	Non—normal	1000 (500,500)	8	2
	Non—linear	1000 (500,500)	2	2
	Iris2	100 (50,50)	4	2
	Ionosphere	351 (176,175)	34	2
Real-life	Sonar	208 (104,104)	60	2
	Arrhythmia	452 (226,226)	279	16
	Glass	214 (107,107)	9	2
	Adult4	8336 (4168,4168)	14	2

Table 3 The prototype reduction rates (%) of the creative PRS on the nine data sets. The results of the CNN are included as a reference. Also, in the interest of clarity, we have distinguished between the 'Artificial' and 'Real Life' data sets, respectively. The first three rows represent the 'Artificial' data sets, and the last six rows represent the 'Real Life' data sets

Data	CNN	PNN	VQ	BT	HYB
	82.00	85.00	36.00	84.00	90.00
Rand	85.00	87.50	36.00	84.00	91.00
	83.50	***86.25**	**36.00**	**84.00**	***90.50**
	87.20	88.80	99.20	96.80	87.80
Non—n	86.80	24.00	99.20	96.80	88.60
	87.00	**56.40**	***99.20**	***96.80**	**87.90**
	80.60	82.60	87.20	93.60	56.20
Non—l	79.60	82.00	87.20	93.60	53.20
	80.10	**82.30**	***87.20**	***93.60**	**54.70**
	70.00	86.00	68.00	92.00	88.00
Iris2	76.00	80.00	68.00	92.00	82.00
	73.00	**83.00**	**68.00**	***92.00**	***85.00**
	71.02	78.86	97.73	81.82	75.00
Ionos	76.14	81.14	97.73	81.82	73.86
	73.58	**80.00**	***97.73**	***81.82**	**74.43**
	50.00	67.31	69.23	38.46	49.04
Sonar	49.04	68.27	69.23	38.46	45.19
	49.52	***67.79**	***69.23**	**38.46**	**47.12**
	85.84	96.46	85.84	43.36	67.26
Arrhy	87.61	96.90	85.84	43.36	64.16
	86.73	***96.68**	***85.84**	**43.36**	**65.71**
	70.09	72.90	85.19	40.19	74.07
Glass	69.16	71.96	85.19	40.19	68.52
	69.63	**72.43**	***85.19**	**40.19**	***71.30**
	81.86	84.21	99.62	99.81	89.66
Adult4	81.93	84.07	99.62	99.81	89.23
	81.90	**84.14**	***99.62**	***99.81**	**89.45**

versally true. However, in Chang [5], it has been suggested that the computation time could be significantly reduced by employing an algorithm similar to the minimal spanning tree, and using a method of associating with every sample point t^i in the data set, two distances w_i and b_i. The table shows that the time of all PRS increases as the number of samples and the dimensions are increased. For low-dimensional applications, the VQ and BT perform better than the others in terms of their speed. On the other hand, for high-dimensional data sets like the 'Arrhythmia', the HYB performs better than the others.

As mentioned in Sect. 2, in the VQ and BT methods, the number of prototypes can be controlled by the algorithms themselves.

Our present results also confirm one of the existing 'conflicting' hypotheses concerning the relation between the accuracy and the reduction rate. One school of thought generally believes that as the number of prototypes increases, the error, $(1 - Acc)$, increases. The other school of thought holds to the view that this is not the case. Our results confirm the latter.

It is also known that for high-dimensional data, a non-parametric procedure needs a large number of samples, in order to reliably estimate the Bayes error. However, Fukunaga [9] and Hamamoto et al [11], reported that in the case of the BT algorithm, only a relatively small number of prototypes are needed to get a higher *Acc*. As opposed to this proposition, Xie et al [10] commented that in the case of the VQ, the *Acc* increases as the number of prototypes is increased. Our results demonstrate that none of these statements are *universally* true ! A comparison of the classification accuracy, *Acc*, with the number

Table 4 The classification accuracy rates (%) of the creative PRS on the nine data sets. The results of the CNN are included as a reference. Also, in the interest of clarity, we have distinguished between the 'Artificial' and 'Real Life' data sets respectively. The first three rows represent the, 'Artificial' data sets, and the last six rows represent the 'Real Life' data sets

Data	CNN	PNN	VQ	BT	HYB
	97.50	96.00	98.00	88.00	96.50
Rand	95.00	95.50	96.00	85.10	98.50
	96.25	**95.75**	***97.00**	**86.55**	***97.50**
	92.60	92.40	95.60	95.60	51.40
Non—n	91.20	91.60	94.80	94.66	40.26
	91.90	**92.00**	***95.20**	***95.13**	**45.83**
	88.40	85.60	91.40	77.50	87.80
Non—l	86.40	85.80	89.40	76.92	86.26
	***87.40**	**85.70**	***90.40**	**77.21**	**87.03**
	94.00	94.00	94.00	90.20	94.00
Iris2	84.00	94.00	98.00	81.20	92.00
	89.00	***94.00**	***96.00**	**85.70**	**93.00**
	85.23	85.71	84.66	79.38	87.95
Ionos	78.41	79.43	86.93	76.76	81.70
	81.82	**82.57**	***85.80**	**78.07**	***84.83**
	78.85	82.69	79.81	77.31	78.85
Sonar	80.77	82.69	78.85	73.08	82.21
	79.81	***82.69**	**79.33**	**75.20**	***80.53**
	95.58	98.67	99.56	99.03	98.85
Arrhy	97.35	99.56	97.35	98.14	98.89
	96.47	***99.12**	**98.46**	**98.59**	***98.87**
	61.68	2.62	74.07	42.59	66.67
Glass	66.36	60.75	64.81	48.43	71.30
	64.02	**61.69**	***69.44**	**45.51**	***68.99**
	91.39	88.84	81.09	93.43	93.15
Adult4	91.77	89.97	80.28	93.80	92.54
	91.58	**89.41**	**80.69**	***93.62**	***92.85**

Table 5 The computation time (in seconds) of the creative PRS on the nine data sets. The results of the CNN are included as a reference. Also, in the interest of clarity, we have distinguished between the 'Artificial' and 'Real Life' data sets, respectively. The first three rows represent the 'Artificial' data sets, and the last six rows represent the 'Real Life' data sets

Data	CNN	PNN	VQ	BT	HYB
	0.01	0.71	0.02	0.05	8.51
Random	0.01	0.68	0.03	0.05	4.69
	0.01	**0.70**	**0.03**	**0.05**	**6.60**
	0.14	23.23	0.06	0.32	4.91
Non—n	0.14	59.71	0.06	0.32	24.33
	0.14	**41.47**	**0.06**	**0.32**	**14.62**
	0.12	10.53	0.23	0.47	2.05
Non—l	0.11	10.13	0.23	0.46	2.81
	0.12	**10.33**	**0.23**	**0.47**	**2.43**
	0.01	0.02	0.04	0.01	0.18
Iris2	0.01	0.02	0.05	0.01	0.14
	0.01	**0.02**	**0.05**	**0.01**	**0.16**
	0.16	2.52	0.19	0.18	0.30
Ionos	0.09	2.34	0.19	0.18	0.22
	0.13	**2.43**	**0.19**	**0.18**	**0.26**
	0.18	1.26	0.84	0.21	1.77
Sonar	0.18	1.38	0.73	0.21	1.90
	0.18	**1.32**	**0.79**	**0.21**	**1.84**
	0.90	33.05	5.37	20.81	3.95
Arrhy	0.77	35.58	5.39	20.79	3.04
	0.84	**34.32**	**5.38**	**20.08**	**3.50**
	0.06	0.62	0.13	0.05	1.17
Glass	0.06	2.91	0.11	0.05	0.91
	0.06	**1.77**	**0.12**	**0.05**	**1.04**
	240.96	16954.75	1.38	52.13	4050.80
Adult4	234.41	16772.52	1.24	51.70	6856.48
	237.69	**16863.64**	**1.31**	**51.92**	**5453.64**

Table 6 A comparison the classification accuracy (%) to the number of prototype vectors in the VQ and BT methods. For each data set, the classification accuracy of the first row is of the VQ and that of the second is of the BT. Again, we have distinguished between the 'Artificial' and 'Real Life' data sets, respectively. The first three rows represent the 'Artificial' data sets, and the last six rows represent the 'Real Life' data sets

Data Set	Methods	# of Prototype vectors							
		2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8
Rand	VQ	84.00	84.50	86.75	94.75	96.25	96.00	97.00	–
	BT	79.88	80.75	81.75	85.00	86.55	86.45	86.40	–
Non—n	VQ	95.00	95.20	94.50	90.90	89.30	91.20	90.30	91.90
	BT	94.08	94.65	94.14	95.02	94.95	94.98	95.00	95.00
Non—l	VQ	57.4	75.80	81.60	85.20	89.20	90.40	88.70	87.70
	BT	57.44	68.36	67.38	75.25	77.21	75.24	74.18	73.59
Iris2	VQ	80.00	93.00	87.00	96.00	92.00	–	–	–
	BT	74.60	85.00	83.00	81.00	82.00	–	–	–
Ionos	VQ	74.15	85.80	83.52	80.11	79.26	80.97	78.98	–
	BT	64.18	68.86	72.59	76.14	78.07	77.47	76.42	–
Sonar	VQ	69.71	67.31	69.23	77.40	79.33	76.44	–	–
	BT	58.27	64.18	63.85	71.44	70.53	72.16	–	–
Arrhy	VQ	96.68	97.13	98.45	98.45	98.46	98.45	97.35	–
	BT	95.05	96.49	96.97	97.48	97.37	97.70	97.86	–
Glass	VQ	63.43	62.50	67.13	69.44	66.21	68.06	–	–
	BT	43.94	39.26	42.60	41.67	44.91	45.51	–	–
Adult4	VQ	82.15	66.53	76.26	80.69	80.34	79.81	75.43	71.18
	BT	91.74	91.71	93.62	91.51	92.31	92.12	89.91	87.80

Table 7 A comparison the computation time (in seconds) to the number of prototype vectors in the VQ and BT methods. For each data set, the computation time of the first row is of the VQ and that of the second is of the BT. Again, we have distinguished between the 'Artificial' and 'Real Life' data sets, respectively. The first three rows represent the 'Artificial' data sets, and the last six rows represent the 'Real Life' data sets

Data Set	Methods	# of Prototype vectors							
		2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8
Rand	VQ	0.06	0.03	0.04	0.07	0.10	0.17	0.26	–
	BT	0.03	0.02	0.02	0.03	0.04	0.06	0.10	–
Non—n	VQ	0.16	0.07	0.09	0.10	0.22	0.36	0.62	1.14
	BT	0.18	0.12	0.19	0.31	0.58	1.09	2.11	4.14
Non—l	VQ	0.14	0.12	0.13	0.14	0.20	1.72	0.33	0.50
	BT	0.04	0.07	0.12	0.24	0.47	0.91	1.80	3.59
Iris2	VQ	0.05	0.02	0.02	0.03	0.03	–	–	–
	BT	0.01	0.01	0.02	0.01	0.02	–	–	–
Ionos	VQ	0.37	0.20	0.26	0.37	0.50	0.74	0.61	–
	BT	0.20	0.23	0.30	0.37	0.58	0.96	1.76	–
Sonar	VQ	0.25	0.71	1.34	1.92	2.64	3.54	–	–
	BT	0.21	0.25	0.26	0.32	0.45	0.69	–	–
Arrhy	VQ	1.58	5.00	9.16	14.72	18.53	24.94	34.62	–
	BT	1.70	2.06	2.73	4.03	6.92	11.89	23.06	–
Glass	VQ	0.09	0.10	0.13	0.12	0.15	0.18	–	–
	BT	0.01	0.01	0.01	0.02	0.03	0.05	–	–
Adult4	VQ	0.48	0.61	0.90	1.31	1.98	3.24	5.72	9.22
	BT	12.28	25.99	51.84	103.54	210.36	420.98	838.67	1681.04

of prototypes, *Re*, in the VQ and BT methods for the 'Arrhythmia' and 'Non—normal' data sets shows that the latter index remains almost the same, or falls marginally, as the number of prototypes is increased. Meanwhile, the value of *Acc* for other data sets such as for the 'Random' and 'Sonar' increases as the number of prototype vectors is increased. As asserted earlier, from these results, we conclude that the relation between *Re* and *Acc* completely depends upon the problem domain, and not on the method *itself*. Hopefully, these results resolve these apparent contradictions.

We have also included in Tables 6 and 7 the results obtained for comparing the classification accuracy and the computation time (in seconds) to the number of prototype vectors in the VQ and BT methods, where the two lines for each data set are the results for the VQ and the BT method, respectively. Tables 6 and 7 show a comparison between the classification accuracies (%) and a comparison between the computation time (in seconds) in the VQ and BT methods.

Ranking the various PRSs

Although we have repeatedly mentioned that it is hard to quantitatively compare the various PRS, to render this

Table 8 A ranking of the creative PRSs in terms of their prototype reduction rates (%), the classification accuracy (%), and the computation time (in seconds) for the nine data sets. Here, the Artificial Set consists of 'Random', 'Non—n' and 'Non—l'. The 'Real-life' Sets (Small) and 'Real-life' Sets (Large) are, respectively, 'Iris2', 'Ionos', 'Glass', 'Adult4', and 'Sonar', 'Arrhy'. Also, the values of (·) of each PRS are the ranking indices obtained from the competition as explained in the text

Dataset Type	Reduction Rates (%)	Classification Accuracy (%)	Computation Time (in sec.)
Artificial Set	VQ(3), BT(3), HYB(2), PNN(1)	VQ(5), HYB(2), BT(1)	VQ(6), BT(3)
Real-life Set (Small)	VQ(6), BT(3), PNN(2), HYB(1)	VQ(6),HYB(3), PNN(3)	BT(8), VQ(3), PNN(1)
Real-life Set (Large)	PNN(2), VQ(2), BT(2)	PNN(2), BT(2), HYB(2)	VQ(3), HYB(2), BT(1)

Table 9 A ranking of the creative PRSs in terms of their prototype reduction rates (%), the classification accuracy (%), and the computation time (in seconds) for the nine data sets. Here, the 'Artificial Sets' consists of 'Random', 'Non—n' and 'Non—l'. The 'Real-life' Sets (Low-dimension) and 'Real-life' Sets (High-dimension) are, respectively, 'Iris2', 'Ionos', 'Glass', 'Adult4' and 'Sonar', 'Arrhy'. Also, the values of (·) of each PRS are the ranking indices obtained from the competition as explained in the text

Dataset Type	Reduction Rates (%)	Classification Accuracy (%)	Computation Time (in sec.)
Artificial Set	VQ(3), BT(3), HYB(2), PNN(1)	VQ(5), HYB(2), BT(1)	VQ(6), BT(3)
Real-life Set (Low-dimension)	VQ(5), BT(5), PNN(1), HYB(1)	VQ(6),HYB(4), BT(2), PNN(1)	BT(8), VQ(3), PNN(1)
Real-life Set (High-dimension)	PNN(3), VQ(3)	PNN(4), HYB(2)	BT(7), VQ(4), PNN(1)

comparative study more complete, we have attempted to do exactly this. To achieve this goal, we have graded every PRS tested in terms of their quality indices, where the latter is either the reduction rate, the accuracy or the time required to yield the corresponding reduced sets. We have then merely given 'grades' for the best *two* algorithms for any specified quality index, and added the ranks of all the algorithms for all the data sets[12].

The tables listing the ranking are given in Tables 8 and 9. Table 8 ranks the algorithms in terms of their prototype reduction rates (%), the classification accuracy (%), and the computation time (in seconds), but the data sets are divided in terms of their size. As opposed to this, Table 9 ranks the algorithms according to the *same* criteria, but the real-life data sets are sub-divided in terms of the dimensionality. To aid in the comparison, the ranking of the artificial data sets is included in both the tables.

From both the data sets, it appears as the VQ algorithm is superior to the others for artificial data, and for small data sets. The PNN seems to be the 'winner' when the data sets are large or of a high dimensionality.

[12] We agree that is a very simplistic model of comparison, but that seems to be the best a researcher can do with data which is so varied, and algorithms which have vastly different parameters. A more scientific model for comparison will be discussed in the next section, and is an avenue for future work.

We conclude this subsection but noting that this comparison must be taken for what it is – namely, one done based on a fairly simplistic model. A proposed strategy for a more scientific comparison follows.

Comparison using meta-classification techniques

Readers will observe that although the comparison between the various algorithms is experimentally verifiable, in the final analysis, no statement asserting the conclusive superiority of one method over another, can be made. This is true not only for the results we have submitted, but also for the work that other researchers have published. To our knowledge, the only work which goes beyond merely examining the methods in terms of their classification but also objectively examines the data sets, is the work due to Sohn [22]. We feel that it is pertinent to spend some time explaining these results, as this is probably the next avenue in which our present research would develop.

Sohn [22] attempted to classify the data sets in terms of various features (which we shall call *meta-features*), and then to specify the superiority of the various algorithms based on these meta-features. Her aim was to quantify the performance of each algorithm based on the meta-data characteristics of the data to be classified. She argued that any data set had some inherent characteristics (such as the correlation, the skew, the kurtosis, etc.), and the intention was to be able to determine which algorithm was to be used if the data being processed had specific meta-data features.

Readers will observe that what Sohn did was to essentially transform the 'Find Best Algorithm' problem into a pattern classification problem, where the latter was achieved on the basis of the meta-data features. To achieve this she related the performance of each algorithm to the characteristics of the data to be classified by developing a statistical meta-model by using 18 data sets and 11 classifiers.

In the statistical meta-model, a log transformation on the classification error was invoked to ensure the fitted rate was within 0 to 1 as follows:

$$\begin{aligned} Y_{ij} &= \ln\{\epsilon_{ij}/(1-\epsilon_{ij})\} \\ &= \beta_{0_j} + \beta_{1_j} x_{i_1} + \ldots + \beta_{m_{i_j}} x_{i_{m_i}} + \epsilon_{ij}, \end{aligned} \tag{1}$$

where ϵ_{ij} is the classification error obtained from data set i, ($i = 1,\ldots,n_j$). and where j is the 'index' of the classification algorithm ($j = 1, \ldots, 11$). In the above, $\{x_{m_i}\}$ are the characteristic meta-features of data set i, and n_j is the number of data sets with all m data characteristics to which each classification algorithm j is applied. Finally, ϵ_{ij} is a noise variable assumed to obey $N(0,var(\epsilon_{ij}))$.

This model was fitted with the 12 data characteristics including the geometric mean ratio of the pooled standard deviations to standard deviations of the individual populations, the mean absolute correlation coefficient between two features, the first canonical correlation between a

linear combination of class variables and a linear combination of the features, the proportion of total variation explained by the first canonical discriminant, the mean skewness of the features, the mean kurtosis of the features, the average entropy of the discrete features, the entropy of the classes, the joint entropy of a class variable and the *l*th attribute, and the mutual information between the class and the feature. Other parameters which were involved in the comparison were the number of features, the number of training samples, and the number of binary feature variables.

The novel characteristic of the paper was that the author was able to determine the best method that could be used in a classification problem by merely studying the meta-features of the classes.

As mentioned earlier, there are indeed, 'zillion' of PRS. No single researcher has been successful in determining which PRS method will be superior for any given data set. Rather, the general conclusion has always been that there is no such 'best' algorithm, and that the final decision of the PRS scheme to be used depends actually on the user's preferences and the criteria for evaluating them. This is also the conclusion of our study. Indeed, it is clear that a comparison of reduction rates alone does not make sense. The reduction rate should be used *together with* the accuracy and the computational time, and not as a separate evaluation criterion.

There is now doubt in our minds that the only way to proceed with a more systematic comparison of the PRS is to resort to a meta-classification strategy as in Sohn [22]. However, the problems that are encountered in designing such a meta-classification strategy are numerous. First, unlike the traditional classification problem, the goal of a PRS is *not merely* to yield the best classification. Rather, the intention is that we are willing to forfeit some accuracy so as to retain the prominent prototypes. A second goal in a PRS is to find the most suitable prototypes in a time-efficient manner. The third goal of a PRS is to see how the prototypes can be obtained *without* processing all of the data involved. The reader will quickly observe that the paper (indeed, the very *problem*) considered by Sohn [22] does not concern itself with *any* of these issues. Thus, as can be observed, Sohn's meta-features [22] have nothing to do with computation time, which subset of the features are used, etc. This renders the two problems vastly different, and shows that the problem of designing a meta-classification scheme to lead to a suitable PRS, is a rich avenue for future research.

Any researcher who attempts to tackle this problem will first have to arrive at a suitable criterion function, which involves the data characteristics, the computation time, the size of the reduced prototype set, and probably the 12 meta-features used by Sohn. Additionally, we believe that the criterion function must also consider *how* the data is partitioned so as to lead to subsets whose meta-features are evaluated. All of these are open problems which we are currently investigating.

[13] We are grateful to the anonymous referee who informed us about the work of Sohn [22].

Conclusions

In this paper, we report the results of conducting a comparative study on the various creative Prototype Reduction Schemes (PRS) such as the PNN, VQ, BT and HYB. These PRS have been tested for three artificial data sets and six real-life benchmark data sets, and compared them with each other from the perspectives of their data reduction rates, their classification accuracy, and their computational efficiencies.

Our experimental results demonstrate that no method can be crowned as the 'best' for all kinds of applications. However, in the interest of completeness, we have graded the various algorithms using a rather simplistic ranking scheme. Although no significant differences between the creative PRS is obvious when both the reduction rate and the classification accuracy are considered, the PNN performs better than the others for high-dimensional applications. On the other hand, for low- and middle-dimensional applications, the HYB and the VQ perform better than the other methods. With respect to the computation time, the VQ and BT perform better than the others in low-dimensional applications. Meanwhile, for high-dimensional data sets, the HYB method is faster than the others.

The experimental results also illustrate that the relationship between the reduction rate and the classification accuracy depends on the problem domain. As the number of prototype vectors is increased, the accuracies increase or decrease *a little* depending on the characteristics of the specific dataset.

We have also discussed a potential avenue for future research. This is based on the work of Sohn [22], and is one that would utilise meta-features to determine the best PRS scheme to be used for any given data set.

Acknowledgements

We are very grateful to the anonymous Referees for their valuable comments, which improved the quality and readability of the paper. In particular, the entire subsections about the criteria for the taxonomy, and the potential for meta-classification in determining the best PRS, was motivated by their comments. The first author is also grateful to the School of Computer Science at Carleton University, Ottawa, for being willing to host him during his year of sabbatical leave.

References

1 Bezdek JC, Kuncheva LI (2000) Nearest prototype classifier designs: An experimental study. International Journal of Intelligent Systems 16(12):1445–1473

2 Dasarathy BV (1991) Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques, IEEE Press, Los Alamitos, CA
3 Hart PE (1968) The condensed nearest neighbor rule'. IEEE Transactions on Information Theory 14:515–516
4 Gates GW (1972) The reduced nearest neighbor rule. IEEE Transactions on Information Theory 18:43–433
5 Chang CL (1974) Finding prototypes for nearest neighbor classifiers. IEEE Transactions on Computers 23(11):1179–1184
6 Ritter GL, Woodruff HB, Lowry SR, Isenhour TL (1975) An algorithm for a selective nearest neighbor rule. IEEE Transactions on Information Theory 21:665–669
7 Tomek I (1976) Two modifcations of CNN. IEEE Transactions on Systems, Man and Cybernetics 6(6):769–772
8 Devijver PA, Kittler J (1980) On the edited nearest neighbor rule. Proceedings 5th International Conference on Pattern Recognition, pp. 72–80
9 Fukunaga K (1990) Introduction to Statistical Pattern Recognition, Second Edition. Academic Press, San Diego, CA
10 Xie Q, Laszlo CA, Ward RA (1993) Vector quantization techniques for nonparametric classifier design. IEEE Transactions on Pattern Analysis and Machine Intelligence 15(12):1326–1330
11 Hamamoto Y, Uchimura S, Tomita S (1997) A bootstrap technique for nearest neighbor classifier design. IEEE Transactions on Pattern Analysis and Machine Intelligence 19(1):73–79
12 Kohonen T (1995) Self-Oganizing Maps. Springer-Verlag, Berlin
13 Song HH, Lee SW (196) LVQ combined with simulated annealing for optimal design of large-set reference models. Neural Networks 9(2):329–336
14 Bezdek JC, Reichherzer TR, Lim GS, Attikiouzel Y (1998) Multiple-prototype classifier design. IEEE Transactions on Systems, Man and Cybernetics 28(1):67–79
15 Kuncheva LI, Bezdek JC (1998) Nearest prototype classification: Clustering, genetic algorithms or random search? IEEE Transactions on Systems, Man and Cybernetics 28(1):160–164
16 Sanchez S, Pla F, Ferri FJ (1997) Prototype selection for the nearest neighbor rule through proximity graphs. Pattern Recognition Letters 18:507–513
17 Lipowezky U (1998) Selection of the optimal prototype subset for 1-NN classification. Pattern Recognition Letters 19(10P:907–918
18 Vapnik VN (1998) Statistical Learning Theory. Wiley, New York
19 Kim S-W, Oommen BJ (2003) Enhancing prototype reduction schemes with LVQ3-type algorithms. Pattern Recognition (to appear)
20 Chen CH, Jowik A (1996) A sample set condensation algorithm for the class sensitive artificial neural network. Pattern Recognition Letters 17(8):819–823
21 Wu Y, Ianakiev KG, Govindraju V (2001) Improvements in k-nearest neighbor classification. Proceedings of ICAPR 2001, LNCS-2013, pp. 222–229.
22 Sohn SY (1999) Meta analysis of classification algorithms for pattern. IEEE Transactions on Pattern Analysis and Machine Intelligence 21(11):1137–1144
23 Linde Y, Buzo A, Gray R (1980) An algorithm for vector quantizer design. IEEE Transactions on Communications 28(1):84–95

Sang-Woon Kim received the BE degree from Hankook Aviation University, Korea in 1978, and the ME and the PhD degrees from Yonsei University, Korea in 1980 and 1988, respectively, both in electronic engineering. In 1989 he joined the Department of Computer Science and Engineering, Myongji University, Korea, and is currently a Full Professor there. From 1992 to 1993, he was a visiting scientist at the Graduate School of Electronics and Information Engineering, Hokkaido University, Japan. From 2001 to 2002 he was a Visiting Professor at the School of Computer Science, Carleton University, Canada. His research interests include Pattern Recognition, Machine Learning and Avatar Communications in Virtual Worlds. He is the author or co-author of 17 regular papers and nine books. He is a member of the IEEE, the IEICE and the IEEK.

John Oommen was born in Coonoor, India on September 9, 1953. He obtained his BTech degree from the Indian Institute of Technology, Madras, India in 1975. He obtained his ME from the Indian Institute of Science in Bangalore, India in 1977. He then went on for his MS and PhD, which he obtained from Purdue University, in West Lafayettte, Indiana in 1979 and 1982, respectively. He joined the School of Computer Science at Carleton University in Ottawa, Canada, in the 1981–82 academic year. He is still at Carleton and holds the rank of a Full Professor. His research interests include Automata Learning, Adaptive Data Structures, Statistical and Syntactic Pattern Recognition, Stochastic Algorithms and Partitioning Algorithms. He is the author of more than 190 refereed journal and conference publications, and is a Fellow of the IEEE. Dr Oommen is on the Editorial Board of the IEEE Transactions on Systems, Man and Cybernetics, and Pattern Recognition.

Originality and contribution

The main contribution of this paper is the formal, objective and systematic study of the various families of creative PRS. Since each researcher attempts to promote his particular paradigm, the literature contains a catalogue of such methods, and various experimental results that arguably demonstrate the superiority of one scheme over the other. The first objective of the paper is to submit a simplified taxonomy of the various PRS available. The primary contribution of this paper is the assertion that there seems to be no clear scheme that is uniformly superior to all other PRS. This has been clearly demonstrated in experiments involving both synthetic and real-life data. Instead, it appears as if each method has its distinct advantages. Furthermore, the question of determining when one method is superior to another, is, generally speaking, open. We believe that this really depends upon the data, its clustering patterns, the outliers of the class distributions, the number of data points themselves, and of course, the proximity relationships between the individual classes.

The results presented here complement the results of the study of previous surveys by authors such as Bezdek. They also form a basis by which some accuracy/reduction-rate hypotheses can be verified. Finally, they can also be used as a benchmark against which future research can be "calibrated".